410-TP-005-001

Distributed Computing Environment Decision within the ECS Project

Technical Paper

December 1996

Prepared Under Contract NAS5-60000

RESPONSIBLE ENGINEER

Shabahat Husain	12/96
Shabahat Husain, Technical Lead EOSDIS Core System Project	Date

SUBMITTED BY

Ramsey Billups /s/	12/96
Ramsey Billups, Release B- Deputy Manager EOSDIS Core System Project	Date

Hughes Information Technology Systems
Upper Marlboro, Maryland

This page intentionally left blank.

Preface

This document describes the Distributed Computing Environment (DCE), multiple cell deployment and migration issues from DCE 1.0.3 to DCE 1.1 for A.x timeframe. As such, it does not require formal Government approval. However, the government reserves the right to request changes on any technical upgrades that may impact the efforts of the ECS Project, within 45 days of initial submittal. Once approved, contractor changes to this document are handled in accordance with EOSDIS Core System (ECS) Project guidelines.

Any questions should be addressed to:

The ECS Project Office
Attn.: Shabahat Husain
Hughes Information Technology Systems
1616 McCormick Drive
Upper Marlboro, Maryland 20774-5372

This page intentionally left blank.

Contents

1. Introduction

2. ECS Overview

3. DCE Multi Cell Deployment

1. Introduction

1.1 Identification

The computing resources of each DAAC need to be linked in such a way that the DAACs autonomy is not compromised. Each one of the DAACs will have its own computing resources along with Xterms, PCs, SUNs, HPs, DECs, SGIs as well as IBM workstations. The DAACs will also have system administrators to manage these computing resources within their own DAAC locations. After conversing with some of the system administrators, it became apparent that the system administrators want to operate their own resources, which provides the advantage of greater autonomy in deciding day-to-day operational policies. There may be some overhead associated with this decision. For example, each one of the DAACs must allocate resources for the DCE Security Server; Cell Directory Services Server (CDS); Time Servers and a DCE trained staff to manage the DAACs. However, the advantages of greater autonomy over some additional cost was preferred by ECS personnel.

While evaluating a single cell approach at the System Monitoring and Coordination Center (SMC) within Goddard Space Flight Center (GSFC), it was realized that ECS operations can not be efficiently controlled from a centralized location. The other disadvantage of the single cell architecture is the single point of failure. Considering the ECS requirements, the SMC cannot control the DAAC's operations. The SMC will only coordinate the activities among different DAACs.

However, if we decide to adopt the multi-cell architecture, it will provide us with scalability. If a DAAC desires to add computer resources to significantly enhance its site, it can do so without affecting other DAAC's operations. If a certain processor at the DAAC (i.e. GSFC DAAC) site is inoperable, the ECS operations at other DAACs can continue avoiding a single point of failure.

1.2 Scope

This DCE multi-cell architecture is in addition to the plan submitted for Release A Critical Design Review (CDR). This DCE paper defines the cell requirements as it relates to the client-server issues at the DAACs and maintenance of Release A and B code compiling requirements as it relates to A.x issues. The scope of this paper will include the migration of Commercial Off-The-Shelf (COTS) software versions (from DCE 1.0.3 to DCE 1.1) and direction on general policies and purposes of the Release A.x DCE multi-cell architecture deployment.

A.x will be the maintenance release that will enable a smooth transition between two major ECS software releases (release A and B). A.x's position within the ECS project includes DCE migration, multiple-cell deployment, new earth science data types (ESDTs) to support SSIT efforts and possibly mode management implementation. In addition, A.x will port Release A software onto Release B platforms.

1.3 Purpose

The purpose of this paper is to present an analysis of Distributed Computing Environment for A.x within the ECS Project in the following areas:

- single cell architecture versus multi-cell architecture;
- flat versus hierarchical cell architecture;
- migration from DCE 1.0.3 to 1.1.

In addition, the DCE paper for A.x will provide direction on scalability, manageability, affiliation and security issues.

1.4 Organization

This document is organized as follows:

Section 1: Introduction- This section presents the document identification, scope, purpose, review and approval, and organization.

Section 2: ECS Overview- This section presents insight into the ECS project.

Section 3: Distributed Computing Environment Multiple Cell Deployment- This section analyzes the decisions associated with single cell versus multi-cell architecture. In addition, this section evaluates scalability, manageability, affiliation and security features for different cell architecture.

Section 4: Migration from DCE 1.0.3 to 1.1- This section compares DCE 1.0.3 to DCE 1.1. It also compares the interface provided by DCE 1.0.3 and supported by DCE 1.1 along with new security features.

Section 5: Conclusion- This section presents the overall recommendations for the DCE activities related to A.x timeframe.

2. ECS Overview

2.1. MTPE and ECS Overview

Mission to Planet Earth (MTPE) is a comprehensive program established by the National Aeronautics and Space Administration (NASA) to study Earth as an integrated and coupled system consisting of the atmosphere, oceans, and continents interacting through exchange of energy, mass, and momentum on a wide range of spatial and temporal scales. The commitment to make data and information resulting from MTPE easily available to users is critical to the success of the project. NASA has started to meet this commitment through incremental and evolutionary development of the Earth Observing System Data and Information System (EOSDIS) with significant user involvement in all of its phases.

2.1.1 EOSDIS Components

EOSDIS is a comprehensive data and information system that must perform a wide variety of functions to support a diverse national and international user community. The key functional requirements to be met by EOSDIS are:

- Planning, scheduling, commanding and controlling EOS spacecraft and instruments
- Data capture and telemetry processing
- Product generation
- Data archival, management and distribution
- Information management
- User support
- System Evolution
- Open architecture
- Transfer of data to long-term archives

The Major components of EOSDIS are:

- EOSDIS Core System (ECS)
- Distributed Active Archive Centers (DAACs)
- Scientific Computing Facilities (SCFs)
- Networks
- EOS Data and Operations Systems (EDOS)

These components are described briefly below.

2.1.1.1 EOSDIS Core System (ECS)

The ECS provides the “core” common capabilities and infrastructure required for planning and scheduling, commanding and controlling, generating products, managing information, archiving and distributing data, and provide access to data held by EOSDIS. The hardware and software developed as a part of ECS resides and operates at the DAACs. ECS consists of three segments: Science Data Processing Segment (SDPS), Communications and System Management Segment (CSMS) and Flight Operations Segment (FOS).

2.1.1.2 Distributed Active Archive Centers (DAACs)

NASA has several DAACs representing a wide range of Earth science disciplines to carry out the responsibilities for processing, archiving, and distributing EOS and related data and for providing a full range of user support. An additional DAAC provides a link between the EOS Program and the socioeconomic and educational user community. The DAACs provide custodianship for the data during the EOS mission and ensure that data will be accessible to users in an user friendly form. These institutions play an active role in the development of EOSDIS both through their own developments and by reviewing the development of ECS through the contractor.

The following DAACs will be participating in the ECS project.

ASF	Alaska SAR Facility
EDC	EROS Data Center
GSFC	Goddard Space Flight Center
JPL	Jet Propulsion Laboratory
LaRC	Langley Research Center
NSIDC	National Snow and Ice Data Center
ORNL	Oak Ridge National Laboratory
SEDAC	Socio-Economic Data and Applications Center

2.1.1.3 Scientific Computing Facilities (SCFs)

The computing facilities used by the EOS investigators (Facility Instrument Team Leaders and Team Members, Instrument Principal Investigators, and Interdisciplinary Investigators) are called SCFs. These facilities range from individual workstation to supercomputers. Having an SCF supported by the EOS Program is not a requirement to be able to access data from EOSDIS or to become a provider of services similar to those in EOSDIS.

2.1.1.4 Networks

The effectiveness of accessing to the data from EOSDIS will depend on the availability of network connectivity between the users and the sources of data. The existing and evolving network capabilities within the United States and abroad will be used to the maximum extent to satisfy the connectivity needs. These capabilities include the NASA Science Internet, its connections to the National Science Foundation (NSF) Internet, and the National Research and Educational Network (NREN) as it develops.

Release A has a single cell architecture, while Release B has implemented a multi- cell architecture. The multiple cells deployment at different DAACs will take place during A.x timeframe.

This page is intentionally left blank.

3. DCE Multi Cell Deployment

3.1 Introduction

This Distributed Computing Environment (DCE) document on multi-cell architecture addresses the set of issues associated with the cell architecture design, advantages and disadvantages of different cell architectures and security schemes. Depending upon the design analysis for different alternatives, schema will be recommended for the cell architecture. The naming, security and other important features will be considered for comparison purposes.

A cell is the basic unit of operation and administration in DCE. It is a group of users, systems, and resources that typically have a common purpose and share common DCE services. At a minimum, the cell configuration includes the Cell Directory Service, the Security Service, and the Distributed Time Service.

A separate cell for the following DAACs are being analyzed.

ASF	Alaska SAR Facility
EDC	EROS Data Center
GSFC	Goddard Space Flight Center
JPL	Jet Propulsion Laboratory
LaRC	Langley Research Center
NSIDC	National Snow and Ice Data Center
ORNL	Oak Ridge National Laboratory
SEDAC	Socio-Economic Data and Applications Center

3.2 Cell Identification

A cell usually consists of nodes in a common geographic area, but geography does not necessarily determine its boundaries. During Release B timeframe, we have two separate cells for SMC as well as GSFC DAAC location. For example, during Release A timeframe, we will have one cell for GSFC and LaRC which are geographically located at two distant locations. It can include one system or as many as several thousand. The cell architecture and its configurations depends on factors such as organizations size, its network topology, and its needs and preferences.

3.3 Single Cell Vs. Multi-Cell Architecture for ECS

Release A is currently using a single cell architecture. However, ECS is a geographically scattered throughout the United States. ECS at post Release A timeframe will consist of multiple DAACs. Each one of these DAACs will have their own computing resources. They will have their Xterms, PCs and workstations. They will also have system administrators to manage these computing resources in their own cell resources. After discussing the DCE management with the system administrators, it became apparent that they want to operate their own DAACs. This will enable system administrators to have a greater autonomy in deciding their day-to-day operation policies. On the other hand, there may be some overhead associated with mutli-cell architecture. For example, each one of these DAACs has to allocate resources for the DCE Security Server, Cell Directory Server and Time Servers and a DCE trained staff to manage it. However, the advantages of greater autonomy over some additional cost is preferred by the ECS personnel.

The following factors were analyzed for the cell architecture design.

3.3.1 Scalability

The cell size should be scalable. There is only 32-bits address space in Security Server and the cds Server. This provides a smaller limit on the number of hosts which can be populated in a single cell. Single cell for multiple-DAACs sites would not leave enough room for the growth. The multi-cell architecture will provide scalability.

3.3.2 Manageability

The multi-cell architecture also provides better manageability. A large single cell architecture may provide a big challenge to cell administrator in managing the entire cell. In addition, a single cell architecture may introduce a single point of failure which reduces the availability of the ECS System.

3.3.3 Affiliation

The trust among users in a local cell is higher as compared to a larger remote cell. The users at a local DAAC have trust and better affiliation among themselves. This makes the system administrator’s tasks easier.

3.3.4 Security

Security issues that help to determine the scope of a cell include: The effect of the security setup on the cross cell communication, the authorization of data for a remote client request and the effect on the delegation chairs requests.

3.3.4.1 Cross-Cell Communication

The following issues are associated with the cross-cell communication in a multi-cell environment. When clients use servers in other cells, the cooperating cells must share a password. The local group membership works fine in a local cell environment. At present, the foreign principals can not be members of a local cell group. On the other hand, if we try to setup the parallel group structure in each different cell then the autonomy is compromised. This issue is being addressed in DCE 1.2.1. Since DCE 1.2.1 will not be available from vendors in Release A.x timeframe, we will not discuss it within this DCE Technical Paper.

3.3.5 Authorization

Because all principals in a cell share a common authentication server and database, a cell should contain a set of users that are more likely to establish trust with each other than with principals outside their cell. The authorization of data in different cells needs to be established based upon the principal and group affiliation. We also know that the larger a cell is, the more work it can be to repair the damage resulting from a breach of security.

Even though ECS has not identified the usage of delegation features at this point. If the delegation features have to be used then different issues related with the single vs. multiple cell architecture and group ACL setup need to be revisited.

Based upon the above factors, the research reflects it is advantageous to use multiple cell architecture for the ECS project.

3.4 Hierarchical versus Flat Cell Architecture

Since we have decided to use the multiple-cell architecture, we need to analyze whether hierarchical cell architecture or flat cell architecture is best.

Advantages of Flat Cell Architecture

- Easy to design
- Simpler architecture

Advantages of Hierarchical Cell Architecture

- Hierarchical architecture needed for the transition has less overhead.
- Smoothes out intercell communication
- Lesser number of keys to be maintained

3.5 Cells and Naming

In the cross-cell communication, the cross cell service location is an important service. The client locates the servers in a foreign cell, using this service. We have a choice to use either Global Directory Service (GDS) X.500 or Domain Name Service (DNS) as the external naming agency. We have decided to use DNS as an external naming agency because of its popular usage.

3.6 Cells and Naming for the cross-cell communication

DCE provides a naming system that is flexible enough to represent any kind of resource shared between host systems and broad enough to make resources addressable from any cell in the world.

Each resource in a DCE cell, such as a person, file, device, and so on, has a unique name that distinguishes it from any other resource in any other interconnected cell. The DCE Directory Service stores the names of resources in the DCE. Resources include things like print servers, application servers, or other DCE services. When given a name, the DCE Directory Service returns the unique network address of the named resource by the Cell Directory Service(CDS) within the DCE, or by Global Directory Service(GDS) outside of DCE. When CDS determines that the name is outside the cell, it passes the name to a global name server outside the cell using an intermediary called a Global Directory Agent(GDA). The GDA, a process(called gdad) running on one or more systems in the cell, enables CDS to access a name in another cell using either of the global naming environments(GDS or DNS). GDA is an independent process that can exist on a system separate from a CDS server. CDS must be able to contact at least one GDA to participate in the global naming environment. CDS locates the hosts running the gdad process by reading an attribute called CDS_GDAPointers on the root directory of the local cell (i.e. /.:).

Figure 1 shows how the GDA helps CDS access names outside of a cell. When CDS determines that a name is not in its own cell, it passes the name to a GDA, which searches the appropriate global naming environment for more information about the name. The GDA can help CDS find names in a cell that is registered in GDS(path 1) or a cell that is registered in DNS(path 2). The GDA decides which global service to use based on whether the name syntax is X.500 or DNS.

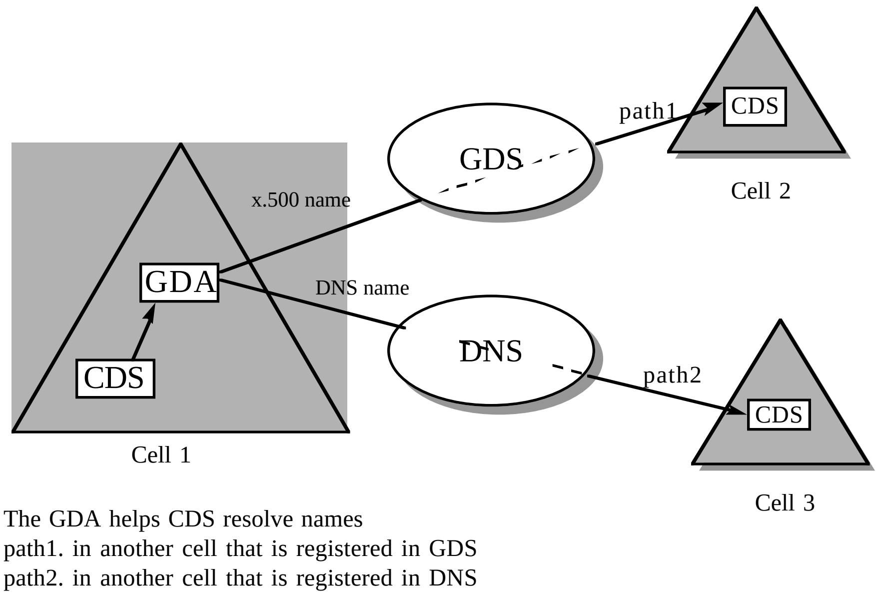

Figure 3-1. Interaction of CDS and A GDA

Figure 2 illustrates how CDS entities are accessed by the clients outside of the local cell.

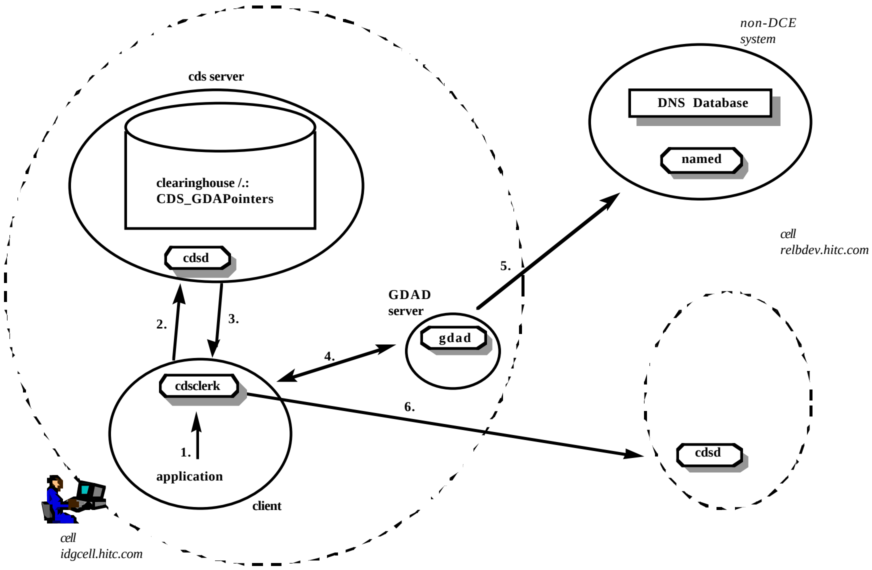

Figure 3-2. Client Request to the GDA

The following six steps are used by a client to locate a server in a foreign cell. In this present example, idgcell.hitc.com is a local cell where as this cell is a foreign cell. The user khill belonging to a foreign cell accesses the server in the remote cell using cross cell location service.

Suppose a client applications makes a request for "/…/relbdev.hitc.com/khill" from the cell idgcell.hitc.com.

The client cdsclerk does not have this cached so it passes the request to the CDS server housing the global root directory.

This server (cdsd) checks the CDS_GDAPointers attribute on "/…" and passes its value back to the client as the place to find a global directory agent.

The cdsclerk uses this value to queries the GDA for the address of a CDS server in the remote call.

The GDA server checks the name syntax to decide whether the name syntax is X.500 or DNS. In this case, it's DNS. The GDA server locates a DNS server via the "resolv.conf" file and the requested cell address is passed to the client.

6. The cdsclerk contacts the cdsd in the remote cell and locates the information requested.

3.7 Release A to A.x/B Cell Transition

Considering all the DCE processes involved in Release A cell to Release B cell transition, we would like to preserve the following important cell assets across the transition, which will entail two major dimensions of change;

- DCE release level 1.0.3 to 1.1
- New (hierarchical) cell name

We would expect to stage the new cell on a new set of Release B machines such that we do not have to bring the release A cell down, in order to begin the transition. The following cell assets need to be preserved across the transition.

1. DCE registry

The user and client machine principals and their passwords as a preference will be preserved. The Application principals and the passwords are not necessary to be preserved at this time because we can configure these new, as we re-stage the applications in the new cell. Since, we need to configure and execute these applications in the new cell after the cell setup, the migration of application principals and the password data is not required.

2. Authorization Data (ACLs)

In Release A the majority of this data is in Sybase database. The fact that Release B will also store this data in Sybase, should greatly facilitate this activity. However, the need to carefully study all the ACLs to ensure that additional links are not imbedded inside the ACLs is needed. It may be discovered that there are imbedded cell names and principals in ACLS. In order to handle some of this data manually i.e. the need may be to perform some uuid substitutions to transition ACLS appropriately.

3. DFS filesets and their associated authorization data

Even though Release A is using DFS on a very limited basis, Release B will be using DFS extensively. DFS also provides extensive utilities for backup and restore of filesets, we should be able to use these facilities to transition DFS files to the new cell.

4. Client machine configurations

Depending on the number of client machines, this may turn out to be the most critical aspect of the transition. What we need is to devise a means to transition a configured host into the new cell, with a minimum of reconfiguration of the client machine.

Please note that CDS migration related issues are not mentioned here because this should essentially be a non-issue. The DCE administrator should set up the new cell name structure and as the new applications are transitioned into the cell, they will export their binding information as appropriate.

During the transition we may want to advertise a cut over date for the subscription data. At the point of the cut over to the new cell, no subscription data will be transitioned.

ESDIS should advise otherwise the subscription and advertisement assets must be preserved across the transition.

Please refer to Appendix A -E which contain the details to facilitate any effort to achieve the objectives outlined above.

3.8 DCE Multi-Cell Deployment

In Release B, we will be migrating from Release A's single- cell architecture to Release B multiple- cell architecture. Once all the applications and Clients have been migrated to Release B Cell, the Release B cell will be considered operational. After the completion of the transition the machines at the Release A cell as well as Release B cell need to be fully operational for some time (the time to be specified later) to ensure that Release A users had enough time for the transition. The Release A cell can be shutdown after the transition period. The Release A cell consists of GSFC, LaRC and SMC DAAC sites. In preparation for the transition the following actions are needed:

- The decision has been made to use the DNS naming instead of X.500 naming convention, the new cell names need to be registered in the DNS. The DNS entry will enable other cells to communicate with the new cells.

- Backup the naming (CDS) and security databases, in case we need to access that information. Note: It is always a good practice to backup the important data.

- All the DCE processes should be brought up as soon as the cell setup is completed. The DCE 1.1 security, naming and time servers need to be brought up to establish a new cell. Since, we have allocated a CSS and a MSS server at each of the DAAC sites, the CSS and the MSS servers should be brought up next.

- Now, one should start populating the DCE 1.1 security database in the new cell. This is somewhat an involved procedure because of DCE 1.1 security considerations with encrypted passwords and the format changes. Hopefully, newly registered users are not logged on at this time, if any new user is logged on, they should be logged out.

- New Access Control Lists (ACLs) should be setup again since, Universal Unique Identifiers (UUIDs) for the principals contained in the ACLs change with the setup of new cells. Once more than two cells have been setup, we need to establish pair-wise key relationships with the security servers of each cell, to allow cross-cell authorization. The new cross cell authorization surrogates should be set up in the release B environment, and no special effort need to be undertaken to preserve the corresponding release A assets.

- After the new Cell is setup properly and servers are running. Gradually, move the old clients to the new cell. The file that specifies the location of the DCE name, security, and time servers need to be modified by running a DCE setup script. If clients are using a configuration parameter specifying the usage of single-cell or multi-cell or site specific names then the configuration parameter need to be changed reflecting multi-cell names. The client should be restarted and tested thoroughly before deploying it in a production environment. Now all the DCE clients at Goddard and LaRC DAAC sites should be operating in their respective cells.

Once GSFC and LaRC cells have been deployed, then the same procedure can be repeated for the SMC site. Once a totally independent cell is deployed for the SMC site then the old and new cells should be running simultaneously for some time. After satisfactory performance of old and new cells for some period of time, the old Release A cell can be shutdown.

3.9 Release B Cell Deployment at New DAACs

The steps for the new Release B Cell Deployment at the new DAACs is quite similar to the steps defined in the above section. However, these steps are outlined here for the sake of completeness.

- Register new cell names in the DNS since the decision has been made to use the DNS naming instead of X.500 naming convention. The DNS entry will enable other cells to communicate with the new Release B cell.

- Prior to setting up the cell, the machines should be configured and identified for the security server, naming server and time servers. The DCE 1.1 security, naming and time servers need to be brought up to establish a new cell. The naming server CDS should be configured using the ECS naming convention.

- Now, one should start populating the DCE 1.1 security database in the new cell. Hopefully, newly registered users are not logged on at this time, if any new user is logged on, they should be logged out. New Access Control Lists (ACLs) should be setup again since Universal Unique Identifiers (UUIDs) for the principals contained in the ACLs change with the setup of new cells. Once more than two cells have been setup, the pair-wise key relationships need to be establish with the security servers of each cell, to allow cross-cell authorization.

This page intentionally left blank.

4. Migration with DCE from 1.0.3 to 1.1

4.1 Commands

In comparing studies on dcecp (DCE 1.1) and cdscp (DCE 1.0.3) command interfaces, we have found that some of the commands are missing and new commands have been added in DCE 1.1. Some of the cdscp commands found in DCE 1.0.3 are missing from DCE 1.1 dcecp commands. During A.x/ B's timeframe, ECS will continue to use both versions 1.1 and 1.0.3. However, there has been some concern that some of the DCE 1.0.3 commands will not be supported from the vendor community in the future.

The comparisons between DCE 1.0.3 and 1.1 are as follows:

4.1.1 DCECP VS. CDSCP Commands

Most of the DCE 1.0.3 cps commands are being supported by DCE 1.1 dcecp commands. However, the following commands are not being supported by dcecp (DCE 1.1) commands.

- Set directory to a new epoch
- Disable Clerk
- Disable Server
- Limited pattern matching

The following dcecp commands have been added in DCE 1.1, which were not there with the cdscp (DCE 1.0.3) commands.

- Directory delete-tree
- Directory merge
- No pattern matching- can use TCL command for pattern matching
- Supports list- based operations

The cdscp contains 47 commands while new commands have been added in DCE 1.1. However, new commands have been added in DCE 1.1. In accordance to OSF (Open Software Foundation), the missing commands which were in DCE 1.0.3 and not in DCE 1.1.may be intended for the later DCE releases or may not be supported at all.

4.1.2 DTS Commands

Distributed Time Server (DTS) offers all features that are normally provided by a time service, but it also has several features that enhance network performance. It provides the following features:

DTS synchronizes the system clocks in a network with each other and in the presence of an external time-provider, to the UTC time standard. Distributed application use synchronized clocks for their applications.

Quantitative Inaccuracy Measurement

DTS uses combined time and inaccuracy measurements from one or several sources to calculate the most accurate new clock settings for client systems.

There have been some major changes in Distributed Time Server Control Program (dtscp) interface. The new support added in DCE 1.1 for the remote DTS management. The **dtsdate** command sets local clock of a system to be the same as the host remote_host, running a dtsd server. The purpose of **dtsdate** to ensure that clock skew is minimized at initial cell configuration or at host instantiation, because it is difficult to start DCE and its components if the skew is too great.

The dtsdate command can be used for adjusting a clock backwards, before DCE is running on a host. Adjusting a clock backwards while DCE is running can cause many difficulties, because security and file system software generally require system time to increase monotonically.

There are other minor differences between DCE 1.1 and 1.0.3. The command **intro.** in DCE 1.0 becomes **dts_intro** in DCE 1.1. The following DTS related commands did not exist in DCE 1.0.3: DTS catalog -to query all DTS components in a cell, clock compare -to compare time between two DTS machines.

4.1.3 RPC Commands

The rpccp contains 18 commands while dcecp uses four objects.

- rpcentry- manages server entries
- rpcgroup- manages RPC groups
- rpcprofile- manages RPC profiles
- endpoint- manages endpoint mappings

There is one hundred percent coverage of DCE 1.0.3 commands in DCE 1.1 commands interface.

RPC enhancements

- DCE RPC provides two administrative facilities, the RPC daemon and the RPC control program. These facilities are superseded by the DCE Host daemon (dced) and the DCE control program (dcecp) for OSF DCE version 1.1.
- RPC Enhancements allowed improvements to RPC throughput by providing access to additional client sockets for times of peak usage, and optimizes RPC runtime packets for transmission and fast transport (e.g., FDDI, satellite).

4.2 DCE Super Daemon

In DCE 1.1, a single super daemon, dced replaces rpc daemon and sec-client daemon is present on all machines in a cell. In addition, it adds some other important capabilities such as server management and remote file management. It also integrates sec-clientd and rpc daemon providing endpoint management and security validation. Management is more efficient by improving configuration, execution and maintaining state of servers by providing remote key and file management. The security is also improved because endpoint operations have associated ACLs.

The super daemon enables complete remote administration of DCE services. This includes startup, shutdown and status queries, as well as secure remote management of per-host security data and cell configuration information.

4.3 DCE Security Features (Audit daemon)

Even in DCE 1.1, auditing is not distributed, Clients of the security audit subsystem send audit events to an audit daemon running on their local machines. Only the DCE Security, DCE Time Service, and the audit subsystem itself generate audit events in DCE 1.1. However, a set of APIs are provided to support the development of new application servers using the audit subsystem.

The dcecp commands are used to manage the audit facility. The audit subsystem uses file based fillers and configuration information. The audit daemon is center pielle of the audit subsystem. It receives audit log requests from its clients, which are DCE servers. The servers generate audit events at various points where meaningful activity occurs. Both security daemon and DTS daemon perform auditing to local files rather than depend upon audit daemon.

The event numbers and events classes used in the filter files are defined in the event class configuration files. All these files are accessible to the audit clients and audit daemon through the audit log AP. The dcecp allows an administrator to create and modify filters, view audit logs, and manage the audit daemon.

4.3.1 New Security Features

Some of the new features of DCE 1.1 improve this security of DCE environments such as password strength, preauthentication, auditing and invalid login protection. Other improvements for the distributed application developments include GSSAPI, and delegation.

Initially, type of required authentication is determined by each principal, group or determined by each principal, group or organization. By assigning instances of the preauthentication requirements. Extend registry attribute (EPA) to these entities, KDC will insist that the principal preauthenticate to a level greater or equal to that indicated by its value. The administrator can configure the invalid attempt management by setting two ERAs, or principals, groups or organizations.

The purpose of this password strength management is to allow an administrator to control the selection criteria of passwords. The DCE 1.1 supports non-trivial password strength checking and automatic password generator.

4.3.2 Improved Security Features

In DCE 1.1 the security has been improved from the following aspects:

1. Security Proxy

   Intermediary servers are able to operate acting as the initiating client while preserving both the client's and servers' identities. They can also access control attributes across chained RPC operations.

2. Auditing

   In new DCE 1.1 version, not only administrators are allowed to track security-related events within DCEUs trusted computing base, also interfaces are provided for incorporating auditing functionality into programs.

3. Extended Generic Security Service Application Program Interface (GSSAPI)

   DCE 1.1 allows non-RPC applications to use security features via the standard GSSAPI and also extends GDS to use DCE security via the GSSAPI.

4. Extended Registry Attributes (ERA)

   ERA provides a way of extending the attributes that are stored in the security registry on behalf of a principal, group or organization. It enables single sign-on across non-UNIX platforms and legacy applications in a secure way of associating additional security information with users and groups.

5. Extended Login Capabilities

   This new feature adopts pre-authentication, password management--using strength testing and machine generation. It provides the access for applications from trusted machines.

6. ACL Manager Library

   A subroutine library is provided for easy and fast to implementing an ACL manager for use with all servers.

7. Group Override

New DCE1.1 allows you to customize the group name mapping from host to host with various operating system conventions.

4.4 DCE 1.0.3 to DCE 1.1 Migration Related Issues:

DCE 1.0.3 and DCE 1.1 security servers can coexist as long as the registry version attribute has not been modified to "secd.dce.1.1"; the DCE 1.1 security servers will behave as if they were DCE 1.0.3 servers.

Once the registry version attribute has been upgraded, though, the DCE 1.0.3 servers will be disabled and will have to be upgraded to DCE 1.1 before they can interoperate again (needless to say, this is a unidirectional migration). See Appendix A, for secd migration procedures.

Similarly, DCE 1.0.3 and DCE 1.1 CDS servers can coexist as long as the directory version attribute for root (/.:) has not been modified to "{CDS_DirectoryVersion 4.0}"; DCE 1.1 directories may be replicated on DCE 1.0.3 CDS servers. However, since the root directory is replicated in all clearinghouses, it will be necessary to upgrade the DCE 1.0.3 machines to DCE 1.1 before this attribute can be upgraded on the root directory of any DCE 1.1 CDS server. See Appendix B, for cdsd migration procedures. See also Appendix C, for specific CDS migration issues related to cell aliases, hierarchical cells, etc. DCE 1.0.3 and DCE 1.1 DTS servers should be able to coexist. Note, however, that, as with security and CDS, migration is a one-way process: due to the change of the ACL manager in DTS, when upgrading from 1.0.3 to 1.1, the first time dtsd starts up it reads the old ACL file and writes it back with the new format. (The old file is saved.) In addition, although DCE 1.1 enhances dtscp to be able to remotely manage time servers, a DCE 1.1 dtscp is unable to manage a DCE 1.0.3 dtsd.

In DCE 1.1, rpcd and sec_clientd have been absorbed into and replaced by the DCE daemon, dced (which also incorporates functionality related to serviceability, server and host management, etc.). Systems running rpcd and dced should interoperate with no difficulty; however, two dced-related issues should be mentioned:

1. In DCE 1.1, the CN RPC protocol uses version 5.1 as a default, but can fall back to version 5.0 if 5.1 fails. If a DCE 1.0.3 machine receives a version 5.1 CN RPC, it will generate an error message. This does not mean, however, that the RPC will fail; the 1.1 system will retry using version 5.0. (This is only a migration issue insofar as these error messages could be misconstrued as indicating failed RPCs.)

2. For DCE 1.1, the core services have been rewritten to use the new serviceability error logging facilities instead of the host's system logging. Applications and administrative scripts which extract DCE messages from system logs will have to be modified (or the serviceability routing redirected) to accommodate this change. (Note, however, that in the interest of preserving compatibility with non-DCE Kerberos implementations, DCE Kerberos errors continue to be captured in the host system log.)

Please see Appendices D and E, for information concerning DFS and application migration issues.

This page intentionally left blank.

5. Conclusion

5.1 Summary

Considering the ECS DAACs operational requirements, we studied different factors associated with the cell architecture design. The comparisons have been done to display the advantages and disadvantages of single cell versus multiple cell architecture in detail. Based upon the analysis, the decision was made to use the multiple cell architecture. The multiple cells can be designed using hierarchical or flat cell architecture. The flat cell architecture provides us with an easier to design, and less complex system. The hierarchical cell architecture provides us with the scalability, trust between peer cell and less maintenance because of smaller number of keys to be shared. So for the above reasons, we recommend usage of the hierarchical cell architecture.

There are some advantages of continuing with the usage of existing DCE 1.0.3 like commands. These advantages include usage of the familiar interfaces. However, it is better to learn and start using new dcecp equivalent commands. During our prototype we found, in most of the cases, there is one to one mapping between DCE 1.0.3 cdscp and DCE 1.1 dcecp commands. However, there are some DCE 1.1 dcecp equivalent commands that are not available. Hopefully, this will be supported in the near future. The new dcecp commands can also be used for the extensions and customizations of the existing applications. The recommendation is to change to dcecp equivalent commands as soon as possible. During Release B time frame, we will continue to use DCE 1.0.3/DCE 1.1 clients and DCE 1.1 servers. Since, there is upward compatibility and all of the vendors are supporting both DCE 1.0.3 and DCE 1.1 commands interface, we don’t see any issues, to be addressed, at the present time in migrating from DCE 1.0.3 to DCE 1.1.

This page intentionally left blank.

Appendix A. Security Daemon (secd) Migration

The following procedure was taken from the release notes to the DCE 1.1 Warranty

Patch release:

You may migrate your security servers to DCE 1.1 binaries over any length of time. When each Warranty Patch secd is restarted, it mimics the operation of the 1.0.3 secd.

After you have migrated each of your security servers to DCE 1.1, you can move your cell's security registry to DCE 1.1 operation by issuing, on the master security server, the following dcecp command:

$ dcecp registry modify -version

The master security server will receive and propagate this operation to all cell security replicas.

You should take the following precautions when shutting down the DCE 1.0.3 servers:

1. Always stop security servers via "sec_admin stop".
2. Backup cell database information before installing DCE 1.1.
3. If you are migrating a single-security-server cell, check that the

   acl on /.:/cell-profile contains the following entry:

   user:dce-rgy:rw-t---

Subsequent migration steps require that at least one secd replica have the ability to modify the cell-profile.

Use the following procedure for migrating security server hosts (masters and replicas) from DCE 1.0.3 to DCE 1.1. This procedure may be done simultaneously on each cell member, or over any period of time:

1. Stop 1.0.3 security server via "sec_admin stop".
2. Stop DCE via "dce_config stop".
3. Install DCE 1.1. Reboot the system after you install DCE 1.1.
4. Run "dced -i" to initialize host-specific databases.
5. Copy /etc/rc.dce to /etc/rc.dce.old. Copy /opt/dcelocal/etc/rc.dce to /etc/rc.dce, and comment the appropriate "daemonrunning" lines for the DCE clients and servers on each system.
6. Start DCE 1.1 via "dce_config start".

7. Examine the range of supported versions for the migrated host via "dcecp -c registry show -replica <replica_name>". The "supportedversions" attribute should show that the server supports both 1.0.3 and 1.1.

Servers migrated by this procedure should be able to coexist with original DCE 1.0.3 replicas, in all possible master/slave combinations. The DCE 1.1 secd will mimic 1.0.3 secd behavior until the cell registry is explicitly moved to DCE 1.1.

After you have migrated each security server host to DCE 1.1 binaries, move your cell forward to DCE 1.1 operation as follows:

1. Run "dcecp -c registry modify -version secd.dce.1.1" on the master security server to move the cell forward to DCE 1.1 security.

2. Verify that the new version has been adopted by using "dcecp -c registry show". The value of the version attribute should now be "secd.dce.1.1".

If you have not updated all 1.0.3 security replicas to DCE 1.1, any original 1.0.3 replicas will be stopped when you move the registry version forward to DCE 1.1. You may wish to verify that any original 1.0.3 replicas are no longer running.

Registry version can only be set forward. If you migrate a security server to DCE 1.1 behavior, you cannot revert that server to 1.0.3 behavior.

Appendix B. CDS Daemon (cdsd) Migration

The following procedure was taken from the release notes to DCE 1.1:

To upgrade a directory's CDS_DirectoryVersion, you add the CDS_UpgradeToattribute = 4.0 to the directory. A skulk will propagate this attribute to all replicas of the directory. The next time the background process runs on those servers (which can be forced with the dcecp clearinghouse verify clearinghouse-name command), if the server is running DCE V1.1, it will upgrade the replica's CDS_ReplicaVersion to 4.0. The next skulk of the directory will detect that all replica versions are at 4.0 and finally upgrade the directory's CDS_DirectoryVersion to 4.0. If you are replicating the directory at a pre-V1.1 CDS server, the upgrade will not occur.

Warning: Once you upgrade the CDS_DirectoryVersion, there is no way to then downgrade, as this is a one-way conversion.

To check to see whether your cell's CDS is running Version 4.0, do the following:

In the cell that you care about (parent and child) run this command:

dcecp> directory show /.:

and look at the CDS_DirectoryVersion attribute. It should be 4.0. If not you can't do cellalias create.

For example, you'll notice it's version 3.0 in this case:

dcecp> directory show /.:

{RPC_ClassVersion {01 00}}

{CDS_CTS 1994-10-19-20:10:49.358197100/08-00-09-25-13-52}

{CDS_UTS 1994-10-19-20:11:23.296903100/08-00-09-25-13-52}

{CDS_ObjectUUID c1376a48-eb80-11cd-aa72-080009251352}

{CDS_Replicas

{{CH_UUID c04f605e-eb80-11cd-aa72-080009251352}

{CH_Name /.../absolut_cell/absolut_ch}

{Replica_Type Master}

{Tower {ncacn_ip_tcp 130.105.5.93}}

{Tower {ncadg_ip_udp 130.105.5.93}}}}

```
{CDS_AllUpTo 0}
{CDS_Convergence medium}
{CDS_InCHName new_dir}
{CDS_DirectoryVersion 3.0}
{CDS_ReplicaState on}
{CDS_ReplicaType Master}
{CDS_LastSkulk 1994-10-19-20:10:49.358197100/08-00-09-25-13-52}
{CDS_LastUpdate 1994-10-19-21:10:54.134215100/08-00-09-25-13-52}
{CDS_Epoch c137664c-eb80-11cd-aa72-080009251352}
{CDS_ReplicaVersion 3.0}dcecp>
```

You can upgrade to 4.0 by doing the following:

```
proc cds_upgrade {} {
 directory show /.:
if {[attrlist getvalues $_r -type CDS_DirectoryVersion] != "4.0"} {
directory modify /.: -add {CDS_UpgradeTo 4.0}
directory synchronize /.:
clearinghouse verify /.:/absolut_ch
directory synchronize /.:
directory synchronize /.:}
cdsalias create [getcellname]
cdsalias set [getcellname]}
```

If a directory's CDS_DirectoryVersion attribute is 4.0, this implies that all CDS servers replicating that directory are running DCE 1.1. This is required on the cell's root directory for cell aliasing and hierarchical cells functionality in CDS. In addition, any DCE 1.1 CDS server will use a new ACL manager that recognizes delegation ACLs on directories whose CDS_DirectoryVersion is 4.0.

By default, the default directory version on newly created directories is 3.0. If you are creating a new cell with DCE V1.1, you should start up the CDS server process (cdsd) with the -v 4.0 option, which will make the default directory version at 4.0. We wish to make this a conscious effort because we do not know if you will be replicating directories at pre-1.1 CDS servers.

Appendix C. CDS Migration Issues

The following issues affecting migration from DCE 1.0.3 to DCE 1.1 were identified in the release notes for the DCE 1.1 Warranty Patch:

Cell aliases, hierarchical cells, and transitive trust currently have the following limitations:

Cell alias creation will fail if a cell includes DCE 1.0.x-based clients. The dcecp cellalias script attempts to update every cell-member host by contacting its DCE host daemon (dced). Once the script detects an error (such as failing on a 1.0.x-based client), it will proceed to undo the alias creation operation for the entire cell.

The dcecp cellalias script could be modified to allow alias creation by skipping 1.0.x-based clients or by continuing on error. Note that 1.0.x-based clients would have no knowledge of any aliases for the cell.

When a cell member starts a DCE 1.1 CDS advertiser process for the first time, the old CDS cache is discarded. Information for locating the cell's CDS server processes is also discarded. If the cell member does not share a subnet with a CDS server, you must manually define the location of a CDS server. Do this as follows:

$ cdscp define cached server <name> tower <protseq>:<ip_addr>

Where:

<name> is the name of a machine that is running a cdsd in your cell. It does not need to be the master, especially if the master is "further" (in network distance) away than another cdsd.

<protseq> is a protocol sequence that you want to contact the cdsd with.

<ip_addr> is the IP address of <name>.

Cell renaming does not work reliably. The "cellalias set" dcecp command has been disabled in the Warranty Patch. A defect (OT 12864) has been opened for this problem.

If you wish to create an alternate cell name, use the "cellalias create" dcecp command. This will create a cell alias name without changing the primary cell name.

The significance of this last issue is that it renders moot a migration issue which had been identified in the release notes to DCE 1.1:

There is an outstanding issue related to hierarchical cell support. If a 1.1 cell changes its primary name, then any 1.0.3 cells that had established intercell authentication with the 1.1 cell (that is, exchanged keys) must change the name of the cell principal of the 1.1 cell to be the new primary name in order for the 1.0.3 cell to communicate with the 1.1 cell.

At present this is a non-issue, as cell renaming is not now supported and, according to the release notes for DCE 1.2.1, will not be in the foreseeable future.

Appendix D. DFS Migration Issues

The following DFS migration issues were identified in the release notes to DCE 1.1:

Pre-1.1 acl_edit clients cannot read or modify an ACL in DFS if the ACL contains a new ACL entry type. An error is returned in that case.

A fileset may not be moved or replicated or dumped and restored from a 1.1 DFS server to a pre-1.1 DFS server if any ACL in that fileset contains a new delegation ACL entry type. See also Section 2.17.2 (``DFS Delegation Backward Compatibility Constraints") for details on a required patch to older DFS servers being provided by OSF to properly implement this backwards compatibility constraint.

The page is left blank intentionally.

Appendix E. DCE Application Migration Issues

The following issues affecting application migration from DCE 1.0.3 to DCE 1.1 were identified in the release notes for DCE 1.1:

1.0.3 dynamically-linked applications should work on a 1.1 machine.

There are three known problems preventing 1.0.3 statistically-linked or dynamically-linked applications from working on a 1.1 machine:

If you have a 1.1 wrapper program (for instance, dce_login) that establishes delegation or ERA information, and starts up a 1.0.3 static application, the 1.0.3 application will share the wrapper program's context, but since it has been linked with a 1.0.3 libdce, it doesn't know about the added ERA/delegation fields and will write over them if the ticket expires and this update is needed. This problem causes the loss of delegation/ERA information.

[Note: This apparently is true not just for "wrapper" programs, but for _any_ statically-linked 1.0.3 application which uses authenticated RPCs.]

Another known problem with static applications is related to the credentials file format. This format has changed since 1.0.3, so any 1.0.3 statistically linked application that share some credentials with a 1.1 application will not only not work, but it will behave unpredictably in some cases.

[Note: The change in credentials file format was undertaken in order to maintain compatibility with the MIT Kerberos implementation.]

There is a constraint for XDS applications that would prevent 1.0.3 dynamically-linked applications from working on a 1.1 machine. 1.0.3 applications will not work with 1.1 because of interface changes due to implementation of authentication. The XDS CONTEXT object now contains two new fields, so that an attempt to bind using an old CONTEXT object will fail.

If you are using sec_acl_XXX functions, you can now get delegation ACL entries that your application does not know how to deal with. 1.0.3 statically-linked applications will never see those entries (because they will bind to the old RDACL interface, which will ``pickle'' the delegation entries into the existing ``extended'' entry); dynamic applications (which will use the new 1.1 libdce) will, and will need to be upgraded to deal with new features. Applications writers are encouraged to convert over to use the new DCE ACL library which makes this transparent.

In 1.0.3 many internal CDS routines were called dns_* and there was a header file (dnsclerk_cds.h) that had #defines that turned cds_* into dns_* so that the routines could be accessed via both dns_* and cds_* names.

In 1.1 many of these routines changed so that the actual routine name is cds_* and the header file supports the dns_* name for backward compatibility. However, this means if you call some of these routines in dynamically-linked applications you must recompile them, because the names that the applications are expecting in the libcds have been changed.

Although the CDSPI was not and still is not a public API, we are aware that some developers use it instead of the supported XDS API. None of the CDSPI functions are affected by these changes. We view these functions as those prototyped in cdsclerk.h.

The following is a summary of the changes.

These are in 1.0.3 but not in 1.1:

cdsNoop

cds_fdflags

cds_pid

dns_cleanup_dnsFlagStat

dns_flags_cleanup

dns_handleReplCon

dns_pass_AttributeContents

dns_pass_AttributeSpecifier

dns_pass_NetworkAddress

dns_pass_Progress

dns_pass_ReplicaPointer

dns_pass_Update

dns_pass_UpdatePacket

dns_pass_WholeEntry

dns_pending_cleanup

dns_read_pop

dns_read_push

dns_reader

dns_reader_cleanup

dns_record__init

dns_send_init

dns_send_mutex

dns_utc_add

dns_utc_equal

dns_utc_greater

dns_utc_less

There are new routines in 1.1. The following are those that changed name from dns_* in 1.0.3 to cds_* in 1.1:

cds_cleanup_dnsFlagStat

cds_flags_cleanup

cds_handleReplCon

cds_pass_AttributeContents

cds_pass_AttributeSpecifier

cds_pass_NetworkAddress

cds_pass_Progress

cds_pass_ReplicaPointer

cds_pass_Update

cds_pass_UpdatePacket

cds_pass_WholeEntry

cds_pending_cleanup

cds_read_pop

cds_read_push

cds_reader

cds_reader_cleanup

cds_send_init

cds_send_mutex

cds_utc_add

cds_utc_equal

cds_utc_greater

cds_utc_less

This page intentionally left blank.

Abbreviations and Acronyms

ACL	Access Control List
CDR	Critical Design Review
CDS	Cell Directory Service
CDSCP	Cell Directory Service Control Panel
COTS	Commercial Off-The-Shelf
DAAC	Distributed Active Archive Centers
DCE	Distributed Computing Environment
DCECP	Distributed Computing Environment Control Program
DNS	Domain Name Service
DTS	Distributed Time Service
DTSCP	Distributed Time Server Control Panel
ECS	EOSDIS Core System
EOSDIS	Earth Observing System Data Information System
FDDI	Fiber Distributed Data Interface
GDA	Global Directory Agent
GDS	Global Directory Service
GSFC	Goddard Space Flight Center
HP	Hewlett Packard
MTPE	Mission to Planet Earth
NASA	National Aeronautics and Space Administration
OSF	Open Software Foundation
PC	Personal Computers
RPC	Remote Process Call
TCL	Tool Command Language
UTC	Universal Coordinated Time
SCF	Scientific Computing Facilities
SMC	System Monitoring and Coordination Center
SSI&T	Science Software Integration & Test

This page intentionally left blank.

References

- Hughes Applied Information Systems (1994) System Design Specification for the ECS Project, Document 207-CD-001, Landover, Maryland. Available via URL http://edhs.1.gsfc.nasa.gov/misc/docsw/docswcat.html.
- Open Software Foundation, "OSF DCE User's Guide and Reference", 1993, Prentice Hall.
- Open Software Foundation, "OSF DCE 1.0 Administration Reference", 1993, Prentice Hall.
- Open Software Foundation, "OSF DCE 1.1 Command Reference", October 24, 1994.
- Open Software Foundation, "DCE 1.1 Features", http://www.osf.org/dce/dce11-features.html, Dec. 1, 1995.
- Ward Rosenberry, David Kenney & Gerry Fisher, "Understanding DCE", 1993, O'Reilly & Associates, Inc.

This page intentionally left blank.